EFFICIENT METHOD AND APPARATUS FOR CONVOLUTION OF INPUT SIGNALS

FIELD OF THE INVENTION

[0001] The present invention generally relates to the convolution of input signals, and more specifically to the implementation of artificial reverberation using Fast Fourier Transform (FFT) convolution methods.

BACKGROUND OF THE INVENTION

[0002] Reverberation is the result of a complicated echo system. A listener in a room hears not only the direct signal from the source, but also other reflected sounds from the walls, floor or some other objects in the room. As shown in FIG. 1, the signal heard by the listener is a summation of all reflected signals.

[0003] The effect of reverberation is a multiplicity of temporally close echoes that are not perceptually separate from one another. FIG. 2 shows the impulse response of the Foellinger Great Hall. From FIG. 2, it can be seen that the peaks for later part of the impulse response are very close, and only few peaks in the earlier part clearly stand out of the response. Based on this characteristic, the reverberation can be separated into two parts. As shown in FIG. 3, those peaks in earlier part are called earlier reflections, and the later part is called late reverberation.

[0004] Artificial reverberators have been used to add reverberation to studio recording in the music and film industry, or to modify the acoustic effect of a listening room. There have been basically two approaches to designing reverberators. The first approach is based on the IIR (Infinite Impulse Response)-recursive networks such as comb filters and all-pass filters, and the second approach is based on FIR (Finite Impulse Response) networks. The IIR-based network has the merit in low complexity, but is often difficult to eliminate unnatural resonance. On the other hand, the FIR-based reverberators, which convolve the input sequence with an impulse response modeling the environment such as a concert hall, are free from the unnatural resonance. However, the high computational complexity due to the long FIR length leads to another concern in real-time applications. For two seconds of impulse response, the length is 88,200 samples in terms of 44,100Hz sampling rate. Using direct convolution to implement the reverberation requires 88,200 multiplications for each sample, or 7.8G multiplications per second for stereo audio.

[0005] The IIR-based approach suitably combines various filter modules such as comb filters, all-pass filters, and low-pass filters to simulate the reverberation effect. Due to the nature of the recursive filters, the complexity is in general lower than the FIR-based approach. However, its quality depends on some detail calibration and it is also difficult to model the existing environment directly.

[0006] The FIR-based approach records the environment response, such as a concert hall or a church, as the impulse response and then applies the direct convolution to have the reverberation effect. The environment response can be recorded from real environment using a loud speaker and microphones. FIG. 2 is an example of environment response. The length of a natural environment response may be varying from 1 to several seconds depending on the size of the room, the material of the walls and other surfaces in the room.

[0007] The direct convolution between input signal $x[n]$ and impulse response $h[n]$ of length L is expressed as

$$y[n] = x[n] * h[n] = \sum_{k=0}^{L-1} x[n-k]h[k] \qquad (1)$$

The implementation of (1) is shown in FIG. 4 and its direct implementation leads to L multiplications per output sample, which is too complicated for reverberation. As mentioned above, by direct convolution, convolving a stereo input signal with impulse response requires 7.8G multiplications per second. This is almost impossible for processors today.

[0008] In addition to the direct convolution methods in the time domain, the FIR-based approach can also be implemented by FFT convolution methods in the frequency domain. By means of fast computation accomplished by FFT, the FFT convolution methods significantly speed up the FIR-based approach.

[0009] There have been some researches trying to reduce the complexity of the FIR-based approach by modifying the impulse response according to perceptual criteria. For example, a perceptual convolution method has been proposed to reduce the number of taps in FIR filters to create reverberation without coloration. This approach tries to change the impulse response in time-domain to reduce the multiplications needed for convolution method. However, the approach can only be applied to direct convolution methods. Therefore, its complexity is still higher than FFT convolution methods.

SUMMARY OF THE INVENTION

[0010] This invention has been made to reduce the complexity of implementing artificial room reverberation using FIR-based approaches. A primary object of the invention is to provide an efficient method for the convolution of input signals. It is also an object of the invention to provide an apparatus and method to reduce the complexity of the reverberators using FFT-based methods and the segmented impulse response of the room environment. Another object is to further reduce the complexity using fast perceptual convolution by truncating the high frequency parts of the segmented impulse response based on perceptual thresholds.

[0011] Accordingly, by extending both overlap-and-add and overlap-and-save methods of block convolution to segmented impulse response of the room environment, fast convolution methods based on FFT are used to speed up the FIR-based approaches in generating artificial reverberation. The present invention first segments an environment impulse response, computes its segmented response frequency spectrum by FFT. The input signal is also segmented and FFT transformed to obtain segmented input frequency samples.

[0012] In one embodiment of the overlap-and-add method, the segmented input frequency samples are multiplied by the frequency samples of each segment of the impulse response. The multiplication output of each segment is inversely transformed by IFFT respectively. The outputs of the IFFT from all the segments are then overlapped and added together to generate the final reverberation signal.

[0013] In an alternative embodiment of the overlap-and-add method of this invention, the segmented input frequency samples are buffered segment by segment and then multiplied by the frequency samples of each segment of the impulse response. The multiplication outputs from all the buffered segments are then summed together. The summation output is inversely transformed by IFFT. The output of the IFFT is then overlapped and added together generate the final reverberation signal.

[0014] In another embodiment of this invention, the overlap-and-save method is applied with segmented impulse response. The input signal is first segmented, overlapped and saved. The overlap-and-save input signal is then FFT transformed to obtain the segmented input frequency samples that are buffered segment by segment and then multiplied by the frequency samples of each segment of the impulse response. The multiplication outputs from all the buffered segments are also summed together. The summation output is inversely transformed by IFFT. By discarding the first segment of the output of the IFFT, the final reverberation signal is obtained.

[0015] According to this invention, a fast perceptual convolution is provided to reduce the computational complexity required by FIR-based reverberators. The conventional perceptual approach tries to change the impulse response in time domain to reduce the multiplications needed for the convolution method. The fast perceptual convolution of this invention is to reduce the multiplications needed in frequency domain for the FFT convolution methods by applying some threshold to truncate the segmented spectrum.

[0016] In the fast perceptual convolution of the present invention, the segmented response frequency spectrum of the impulse response is truncated based on a threshold in quiet which is the threshold characterizing the minimum amount of energy needed in a pure tone detected by human hearing system in a noiseless environment. The high frequency parts of the impulse response that are not perceptible are eliminated. The truncated frequency spectrum of the impulse response can then be applied to various embodiments of the invention to further reduce the computational complexity.

[0017] The foregoing and other objects, features, aspects and advantages of the present invention will become better understood from a careful reading of a detailed description provided herein below with appropriate reference to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0018] FIG. 1 shows that a listener in a room hears the signal which is a summation of all reflected signals.

[0019] FIG. 2 shows the impulse response of Foellinger Great Hall.

[0020] FIG. 3 shows a direct signal, early reflections and late reverberation.

[0021] FIG. 4 shows the block diagram of direct convolution for implementing an FIR.

[0022] FIG. 5 shows the block diagram of FFT convolution for overlap-and-add method according to Algorithm 1 of the present invention.

[0023] FIG. 6 shows the block diagram of FFT convolution for overlap-and-add method according to Algorithm 2 of the present invention.

[0024] FIG. 7 illustrates the complexity of Algorithm 1 and Algorithm 2 by means of the number of real multiplications per sample with respect to the block length.

[0025] FIG. 8 shows the block diagram of FFT convolution for overlap-and-save method according to Algorithm 1 of the present invention.

[0026] FIG. 9 shows the block diagram of zero-delay fast convolution implementation for 88200 (90112) samples of impulse response.

[0027] FIG. 10 shows the block diagram of 2-level zero-delay fast convolution implementation of 88200 (90112) samples of impulse response.

[0028] FIG. 11 shows the spectrum of the impulse response recorded from St. John Lutheran Church.

[0029] FIG. 12 shows the spectrum of the impulse response recorded from St. John Lutheran Church after applying the perceptual threshold according to the present invention.

[0030] FIG. 13 shows the block diagram of FFT convolution for overlap-and-add method according to Algorithm 2 of the present invention using fast perceptual convolution.

[0031] FIG. 13A shows the block diagram of FFT convolution for overlap-and-add method according to Algorithm 2 of the present invention with the perceptual sparse processing implemented after the FFT of the input signals.

[0032] FIG. 14 shows the cutoff frequency point found in each block of four different impulse responses.

[0033] FIG. 15 shows the comparison of complexity of fast perceptual convolution and Algorithm 2 when the length of the impulse response is 2 seconds.

[0034] FIG. 16 shows that the fast perceptual convolution can reduce about 30% complexity as compared with Algorithm 2 in real applications.

[0035] FIG. 17 shows the block diagram of the low-latency implementation using fast perceptual convolution according to the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0036] In contrast to direct convolution, a much more efficient approach for implementing the FIR-based methods is to compute convolution through block convolution, in which the signal and impulse response are segmented into sections of length N. Convolution of each block convolution is then implemented through the FFT. There have been two approaches to block convolutions. One is overlap-and-add method and the other is overlap-and-save method. In both overlap-and-add and overlap-and-save methods, the convolution of each pair of small blocks can be accomplished by transforming them from time domain to Discrete Fourier Transform (DFT) domain and performing multiplications on DFT domain. Because the complexity of specific sizes of DFT can be reduced from $O(N^2)$ to $O(N\log N)$ by FFT algorithms, using these algorithms to perform the convolution can significantly reduce the complexity.

[0037] For overlap-and-add method, the convolution is done on each input segment. If the input segment size is N and the impulse response length is L, it will produce $N+L-1$ samples of output for each segment. The later $L-1$ samples of each output segment will affect its following output segments. For each small segment $x_r[n]$ with length N, the convolution produces the corresponding output segments $y_r[n]$ of length $N+L-1$. Then, those output segments are added to produce the result signal $y[n]$. This result is equivalent to the result produced by direct convolution.

[0038] Because the length of the impulse response for room reverberation can be as high as several seconds, the extension of the segmentation can be applied to the impulse response to have the computation merit. To extend the overlap-and-add approach to segmented impulse response, let the input signals $x[n]$ and impulse response $h[n]$ be segmented as a sum of shifted finite-length segments of length *N*, i.e.,

$$x[n] = \sum_{r=0}^{\infty} x_r[n - rN], \tag{2}$$

and

$$h[n] = \sum_{s=0}^{M-1} h_s[n - sN], \tag{3}$$

where *M* is the smallest integer larger than *L* divided by *N*, i.e. $M = \left\lceil \frac{L}{N} \right\rceil$

$$x_r[n] = \begin{cases} x[n + rN], & 0 \le n \le N-1 \\ 0, & \text{otherwise} \end{cases} \quad , \tag{4}$$

and

$$h_s[n] = \begin{cases} h[n + sN], & 0 \le n \le N-1 \\ 0, & \text{otherwise} \end{cases} \quad . \tag{5}$$

Substituting (2) and (3) into (1) yields

$$y[n] = \left\{ \sum_{r=0}^{\infty} x_r[n - rN] \right\} * \left\{ \sum_{s=0}^{M-1} h_s[n - sN] \right\} \tag{6}$$

Because convolution is linear time-invariant, it follows that

$$y[n] = \sum_{r=0}^{\infty} \sum_{s=0}^{M-1} x_r[n - rN] * h_s[n - sN] = \sum_{r=0}^{\infty} \sum_{s=0}^{M-1} y_{r,s}[n - rN - sN], \tag{7}$$

where

$$y_{r,s}[n] = x_r[n] * h_s[n] \qquad \text{for } 0 \le n < 2N - 1 \;. \tag{8}$$

[0039] The convolution of each pair of input signal segment $x_r[n]$ and impulse response segment $h_s[n]$ can be implemented by FFT with $2N-1$ points. For simplicity, the complexity evaluation described here is based on radix-2 FFT and $2N$-point FFT instead of $(2N-1)$-point FFT. Let

$$\hat{x}_r[n] = \begin{cases} x[n+rN], & 0 \le n \le N-1 \\ 0, & N-1 < n \le 2N-1 \end{cases} , \tag{9}$$

and

$$\hat{h}_s[n] = \begin{cases} h[n+sN], & 0 \le n \le N-1 \\ 0, & N-1 < n \le 2N-1 \end{cases} . \tag{10}$$

Because the convolution in time domain is equivalent to the multiplication in frequency domain, (8) can be written as

$$Y_{r,s}[k] = X_r[k] \cdot H_s[k]; \quad for\ 0 \le k < 2N \quad , \tag{11}$$

where $Y_{r,s}[k]$, $X_r[k]$, and $H_s[k]$ are the $2N$-point FFT of $y_{r,s}[n]$, $\hat{x}_r[n]$ and $\hat{h}_s[n]$, respectively.

[0040] According to the above derivation, a fast algorithm is summarized as Algorithm 1 as follows:

Step 1: Store the FFT data of the segmented impulse response, $H_s[k]$.

Step 2: Execute $2N$-point FFT on the segmented input signals to obtain $X_r[k]$.

Step 3: Multiply M pairs of FFT data according to (11). The number of multiplications and additions for each input sample are $2M$ and 0, respectively. Because the input signal and the impulse response are both real signals, the negative frequency part data are the complex conjugate of the positive frequency part. By this property, only $N+1$ multiplications for each block are calculated. This reduces the number of multiplications for each input sample to $M+M/N$.

Step 4: Perform M times the inverse FFT to have the segmented data $y_{r,s}[n]$ for different s.

Step 5: Overlap and add all the segmented $y_{r,s}[n]$ to have the final $y[n]$ according to (7). The number of additions is $2(M-1)$ for each input sample.

[0041] The number of complex multiplications needed per input sample is $(1+M)\mathrm{FFT}(2N)/N+M+M/N = (1+M)(\log_2 N+1)/2-1/N+M$. The algorithm has reduced the complexity of multiplications from L to $2(1+M)(\log_2 N+1)-4/N+4M$. The block diagram for this algorithm is shown in FIG. 5.

[0042] With reference to FIG. 5, the input signal $x[n]$ is segmented by a segment processing unit 501. An FFT processor 502 transforms the segmented signal to frequency samples $X_r[k]$. Frequency samples of the segmented impulse response $H_s[k]$ are stored in the memory blocks 503. The frequency samples of the segmented signal are multiplied by frequency samples $H_s[k]$ of the segmented impulse response in the multipliers 504. IFFT processors 505 then performs inverse FFT. The outputs of IFFT processors 505 are then overlapped and added by means of the adders 506 and buffers 507 to generate the final output signal $y[n]$.

[0043] To reduce the complexity of Algorithm 1, the order of calculations in Algorithm 1 can be changed. Let $p=r+s$, (7) is rewritten as

$$y[n]=\sum_{p=s}^{\infty}\sum_{s=0}^{M-1} y_{p-s,s}[n-pN]=\sum_{p=s}^{\infty}\sum_{s=0}^{M-1} x_{p-s}[n-(p-s)N]*h_s[n-sN]. \qquad (12)$$

Define

$$y_p[n] = \sum_{s=0}^{M-1} y_{p-s,s}[n-pN] = \sum_{s=0}^{M-1} x_{p-s}[n-(p-s)N] * h_s[n-sN] \qquad (13)$$

Hence,

$$y[n] = \sum_{p=s}^{\infty} y_p[n] \qquad (14)$$

The nonzero values of $y_p[n]$ is only in the time interval [pN, pN+2N−2]. Let $n' = n - pN$, equation (13) can be rewritten as

$$y_p[n'+pN] = \sum_{s=0}^{M-1} y_{p-s,s}[n'] \qquad (15)$$

Performing 2N-point FFT on (15) within the nonzero interval [0, 2N−1] leads to

$$Y_p[k] = \sum_{s=0}^{M-1} Y_{p-s,s}[k] = \sum_{s=0}^{M-1} X_{p-s}[k] H_s[k] \qquad \text{for } 0 \le k < 2N-1 \qquad (16)$$

[0044] The fast convolution, referred to as Algorithm 2, is summarized as follows:

Step 1: Store the FFT data of the segmented impulse response, $H_s[k]$.

Step 2: Execute 2N-FFT on the segmented input signals to obtain $X_r[k]$.

Step 3: Multiply and add the two FFT data according to (16). The number of multiplications and additions is both M+M/N for each input sample.

Step 4: Perform inverse FFT to have the segmented data $y_p[n]$.

Step 5: Overlap and add all the segmented $y_p[n]$ to have the final $y[n]$ according to (14). The overlapping factor is 1 and hence has the complexity one.

[0045] The block diagram of the fast convolution is illustrated in FIG. 6. The complexity of multiplications in Algorithm 2 is 2FFT($2N$)/N+M+M/N, which has a factor of up to M times reduction compared to Algorithm 1.

[0046] With reference to FIG. 6, the input signal $x[n]$ is segmented by a segment processing unit 601. An FFT processor 602 transforms the segmented input signal to frequency samples $X_r[k]$. Frequency samples of segmented impulse response $H_s[k]$ are stored in the memory blocks 603. The frequency samples of the segmented input signal are buffered by the buffering units 604 and multiplied by frequency samples $H_s[k]$ of the segmented impulse response in the multipliers 605. The outputs of the multipliers 605 are added together in the summation unit 606. An IFFT processor 607 then performs inverse FFT on the output of the summation unit 606. The outputs of IFFT processors 607 are then overlapped and added by means of adder 608 and buffer 609 to generate the final output signal $y[n]$.

[0047] FIG. 7 illustrates the complexity of Algorithm 1 and Algorithm 2 using the number of real multiplications per sample with respect to the block length. When the input block size is set to 4096, Algorithm 2 needs about 150 real multiplications to convolve a signal with 88,200 samples of impulse response.

[0048] The overlap-and-save method is very similar to the overlap-and-add method except that the input blocks are overlapped, and the output blocks are not overlapped. In the overlap-and-save method, for each input block with a size N, the N samples are combined with the previous $L-1$ samples to form an overlapped input block with $N+L-1$ samples. Then circular convolution or linear convolution is performed on each overlapped input block. The first $L-1$ samples of each output block are discarded. If linear convolution is used, the tailing $L-1$ samples of each output block are also discarded. Finally, the output blocks are concatenated to form the result output.

[0049] To extend the overlap-and-save method to the segmented impulse response, the output signal in (7) is segmented by changing the parameter $r' = r + s$:

$$y[n] = \sum_{r'=0}^{\infty} \sum_{s=0}^{M-1} y_{r'-s,s}[n - r'N] . \qquad (17)$$

Define

$$y'_{r'}[n - r'N] = \sum_{s=0}^{M-1} y_{r'-s,s}[n - r'N] , \qquad (18)$$

where

$$y_{r'-s,s}[n] = x_{r'-s}[n] * h_s[n] \qquad \text{for } 0 \le n < 2N - 1 . \qquad (19)$$

(17) can be represented as

$$y[n] = \sum_{r'=0}^{\infty} y'_{r'}[n - r'N] , \qquad (20)$$

where $y'_{r'}[n - r'N]$ is the summation of all blocks in time interval $[r'N, (r'+2)N-1]$. The form required in the overlap-and-save method should be to separate the output into the non-overlapping blocks $y_r[n]$; that is,

$$y[n] = \sum_{p=0}^{\infty} y_p[n - pN] , \qquad (21)$$

where

$$y_p[n] = \begin{cases} y[n+pN], & 0 \le n \le N-1 \\ 0, & \text{otherwise} \end{cases} \tag{22}$$

[0050] Substituting (20) into (22) yields

$$y_p[n] = \sum_{r'=0}^{\infty} y'_{r'}[n + pN - r'N], \qquad 0 \le n \le N-1 \tag{23}$$

Because each $y_{r'}[n - pN - r'N]$ represents the values at time interval $2N$, there is only two terms in the intervals $[0, N-1]$; that is

$$y_p[n] = y'_{p-1}[n+N] + y'_p[n], \qquad 0 \le n \le N-1 \tag{24}$$

Substituting (18) and (19) into (24) yields

$$\begin{aligned} y_p[n] &= \sum_{s=0}^{M-1} x_{p-s-1}[n+N] * h_s[n] + \sum_{s=0}^{M-1} x_{p-s}[n] * h_s[n], \qquad 0 \le n \le N-1 && (25) \\ &= \sum_{s=0}^{M-1} \{x_{p-s-1}[n+N] + x_{p-s}[n]\} * h_s[n] \qquad 0 \le n \le N-1 && (26) \end{aligned}$$

Let

$$x'_p[n] = x_{p-1}[n+N] + x_p[n], \qquad -N \le n \le N-1 , \tag{27}$$

where $x'_p[n]$ is p-th overlapping block of the input signal $x[n]$. Then, (26) can be rewritten as

$$y_p[n] = \sum_{s=0}^{M-1} x'_{p-s}[n] * h_s[n], \qquad 0 \le n \le N-1 \tag{28}$$

[0051] From (28), each non-overlapping output block can be calculated by evaluating the convolution for overlapping input blocks in the corresponding time interval. The implementations of algorithms described in the previous sections are also applicable to using overlap-and-save method. Algorithm 2 can be modified to use overlap-and-save method as following steps:

Step 1: Store the FFT data of the segmented impulse response, $H_s[k]$.

Step 2: Execute $2N$-FFT on the overlap-segmented input signals to obtain $X'_p[k]$.

Step 3: Multiply and add the two FFT data according to (16). The number of multiplications and additions is both $M+M/N$ for each input sample.

Step 4: Perform inverse FFT to have the segmented data $y_p[n]$.

Step 5: Discard the first N samples of $y_p[n]$ to have the final $y[n]$ according to (28).

The block diagram of the fast convolution is illustrated in FIG. 8. The complexity of multiplications is the same as Algorithm 2.

[0052] With reference to FIG. 8, the input signal $x[n]$ is segmented and overlapped by segment buffers 801 and 802. An FFT processor 803 transforms the segmented signal to form overlapped-and-segmented frequency samples $X'_p[k]$. Frequency samples of segmented impulse response $H_s[k]$ are stored in the memory blocks 804. The frequency samples of the segmented input signal are buffered by the buffering units 805 and multiplied by frequency samples $H_s[k]$ of the segmented impulse response in the multipliers 806. The outputs of the multipliers 806 are added together in the summation unit 807. An IFFT processor 808 then performs inverse FFT on the output of the summation unit 807 to generate the segmented data $y_p[n]$. The first N samples of $y_p[n]$ are discarded in the signal discarding unit 808 to output the final signal $y[n]$.

[0053] Because the block size affects the latency of the system, it is important to shorten the block size to reduce the latency of the system although shortening the block size increases the complexity of the system. For efficiency, the block size is increased to an acceptable range to reduce the complexity. The acceptable latency in applications is about 150 ms which means about 6K samples in terms of 44,100 Hz sampling rate. From FIG. 7, the number of multiplications per sample needed by Algorithm 2 is more than 400 when the block size is set to 1024 samples. To find out the optimal block size, the minimum value of the complexity equation of Algorithm 2 is analyzed as follows.

[0054] From the previous discussion, it is known that the number of complex multiplications per sample is 2FFT(2*N*)/*N*+*M*+*M*/*N*. It is also known that for *N*-point real FFT, the number of complex multiplications needed is (*N*/4)(log$_2$*N* + 3) −1. let *M* be approximated as *L*/*N*. The complexity equation is

$$C(N) = \log_2 N + 4 + (L-2)N^{-1} + LN^{-2} . \tag{29}$$

Differentiating C(*N*) with respect to *N* leads to

$$C'(N) = \frac{1}{N \ln 2} - (L-2)N^{-2} - 2LN^{-3} . \tag{30}$$

The optimum block length N_{opt} can be obtaining through C'(*N*) = 0; that is

$$\frac{N_{opt}^{\ \ 2}}{\ln 2} - (L-2)N_{opt} - 2L = 0 . \tag{31}$$

Hence

$$N_{opt} = \left[L - 2 + \sqrt{(L-2)^2 + \frac{8L}{\ln 2}} \right] \cdot \frac{\ln 2}{2} . \tag{32}$$

[0055] In other words, the block length with best computation efficiency can be obtained if the filter length or the reverberation length is known. For example, when *L* =

88200, $N_{opt} \approx 61140$. N should be limited to be the power of two and the most typical reverberation length is in the range of 2-3 seconds. Another important issue is that the length of the filter is directly proportional to the block length. Furthermore, from FIG. 7, the complexity reduction ratio for N above 4000 is less than 10%. Hence, a value of 4096 for N is a good tradeoff for most environments.

[0056] Because the FFT needs to accumulate a segment to begin the FFT computation, the FFT-based convolution introduced an additional algorithm delay or latency by one FFT block, i.e., N. In some real-time applications like interactive environment, the latency should be limited. In the literature, there have been methods developed to shorten the latency of the filter by using time domain filter with low latency to compute the output of the first impulse response segment.

[0057] To remove the latency of the FFT-based convolution filters, they can be modified by combining with direct convolution to remove the latency. This invention also provides a method to remove the latency of Algorithm 2 so that the demand on the processor is uniform over time.

[0058] Considering Algorithm 2, to shorten the latency, direct convolution is used to calculate the output segment of the first impulse response segment. From (25), the output segment $y_p[n]$ can be expressed as

$$\begin{aligned} y_p[n] &= \sum_{k=0}^{N-1} x[n+pN-k]h[k] \\ &+ \sum_{s=1}^{M-1} x_{p-s-1}[n+N] * h_s[n] + \sum_{s=1}^{M-1} x_{p-s}[n] * h_s[n] \end{aligned} \quad (33)$$

For the first sample of $y_p[n]$, $y_p[0] = y[pN]$, the inputs of the computation are $x_k[n]$, $p-1$

$\geqq k \geqq p - M + 1$ and $x[n]$, $pN \geqq n \geqq pN - N + 1$. The computation of $\sum_{s=1}^{M-1} x_{p-s-1}[n+N] * h_s[n]$ is completed while computing $y_{p-1}[n]$ if the overlap-and-add method is used. Because these inputs are already available when $x[pN]$ is received, $y_p[0]$ can be calculated without waiting for any other input samples and so are other samples in $y_p[n]$.

[0059] Although the implementation of (33) can remove the latency, the computation of $x_{p-1}[n]*h_1[n]$ can only be calculated after the sample $x[pN-1]$ including the last sample of $x_{p-1}[n]$ is available. If the application is to be without any latency, the computation has to be completed in a sampling period. This causes the demand on the processor to become non-uniform over time. To make the demand on the processor uniform, the direct convolution to calculate the output of the first two segments of impulse response can be used. Thus (33) can be expressed as

$$\begin{aligned} y_p[n] &= \sum_{k=0}^{2N-1} x[n + pN - k]h[k] \\ &+ \sum_{s=2}^{M-1} x_{p-s-1}[n+N] * h_s[n] + \sum_{s=2}^{M-1} x_{p-s}[n] * h_s[n] \end{aligned} \tag{34}$$

After this modification, the computation of FFT convolution can be finished in an input segment of time, just like the original algorithm.

[0060] It is known that the direct convolution of N-point impulse response needs N multiplications for each output sample. Thus, after this modification the computational power requirement increases. For example, using Algorithm 2 with 4,096 block size for 88,200 samples of impulse response, it originally takes about 100 multiplications to compute an output sample. After this modification, it may take more than 8,000 multiplications to calculate an output sample. FIG. 9 shows the block diagram of the zero-delay fast convolution implementation for 88,200 (90,112) samples of impulse response.

[0061] To reduce the complexity of the implementation shown in FIG. 9, the block size can be reduced. The complexity equation of the zero delay implementation can be expressed as

$$C_{ZD}(\mathrm{N}) = 4\log_2 N + 16 + 4(L - 2N - 2)N^{1} + 4(L - 2N)N^{-2} + 2N \qquad (35)$$

From (54), it can be found that the optimal block size is 512, and the complexity is about 1760 multiplications per sample.

[0062] Another method to reduce the complexity is that the output of the first 2 segments of impulse response can be calculated with a smaller block size. As shown in FIG. 10, the first two segments are computed with a 256 point direct convolution and a 7936 point fast convolution which has a block size of 128. The other segments are still computed with a block size of 4096. With the implementation of FIG. 10, the complexity is reduced from more than 8000 to about 700 multiplications per sample.

[0063] According to this invention, a fast perceptual convolution is provided to reduce the computational complexity required by FIR-based reverberators. The conventional perceptual approach tries to change the impulse response in time domain to reduce the multiplications needed for the convolution method. The fast perceptual convolution of this invention is to reduce the multiplications needed in frequency domain for the FFT convolution methods by applying some threshold to truncate the segmented spectrum.

[0064] A threshold in quiet is the threshold that characterizes the minimum amount of energy needed in a pure tone detected by human hearing system in a noiseless environment. For the FFT-based method in the present invention, the segmented spectrum $H_s[k]$ can be truncated by comparing the result with the threshold derived from the threshold in quiet. The approach can reduce the complexity required in the FFT-based method. FIG. 11 illustrates the magnitude response of $H_s[k]$ with respect to k and s, it can be seen that the higher frequency part decays faster than the lower frequency part. After partitioning the impulse response, the magnitude of the higher frequency part of later blocks is very small. FIG. 12 illustrates the same magnitude response after applying the threshold in quite to cut the correspondent spectrum lines.

[0065] Considering (16), the output signal $Y_p[k]$ will not be perceptible if the energy is lower than the threshold in quiet. That is

$$\left|Y_p[k]\right| \le Th[k]. \tag{36}$$

where $Th[k]$ is the threshold in quiet for a frequency k. Substituting (16) to (36) leads to

$$\left|\sum_{s=0}^{M-1} X_{p-s}[k]H_s[k]\right| \le Th[k], \text{ for } 0 \le k < 2N-1. \tag{37}$$

Assuming that the signal magnitude is lower than ρ, (37) is reduced to

$$\begin{aligned}&\left|\sum_{s=0}^{M-1} X_{p-s}[k]H_s[k]\right| \\ &\le \rho\left|\sum_{s=0}^{M-1} H_s[k]\right| \le Th[k], \quad \text{for } 0 \le k < 2N-1\end{aligned} \tag{38}$$

The sufficient condition for the above inequality on $|H_s[k]|$ is

$$|H_s[k]| \leq \frac{Th[k]}{M\rho}, \quad \text{for } 0 \leq k < 2N-1 \tag{39}$$

[0066] To implement the fast perceptual convolution, it is necessary to decide the frequency part that can be removed. In Step 1 of Algorithm 1 or 2, the frequency domain data of each small block in the impulse response can be obtained. For each small block, the magnitude of each frequency sample is calculated. Then, the highest frequencies are scanned to find a frequency point in which its magnitude is equal or greater than the perceptual threshold. In Step 3 of both algorithms, the multiplications for those frequencies that are higher than the frequency point corresponding to each block found in Step 1 can be ignored. The block diagram of fast perceptual convolution is shown in FIG. 13.

[0067] FIG. 13 illustrates how the fast perceptual convolution is applied to the fast convolution algorithm, i.e., Algorithm 2 shown in FIG. 6. As shown in FIG. 13, the perceptual sparse processing units 1101 first removes the higher frequency parts of the segmented spectrum $H_s[k]$ that are not perceptible. Once the segmented spectrum $H_s[k]$ is truncated as $H'_s[k]$, the remaining processing is identical to what is shown in FIG. 6. Although no block diagrams are shown to illustrate the application of fast perceptual convolution to the algorithms illustrated in FIG. 5 and FIG. 8, it is clear that perceptual sparse processing units can also be added to them for truncating the segmented spectrum $H_s[k]$ that are not perceptible.

[0068] FIG. 14 shows the cutoff frequency point found in each block of 4 different impulse responses. For those impulse responses, more than 50% of multiplications in frequency domain has been eliminated. For some blocks, the multiplications for the whole block can be removed. FIG. 12 shows the same impulse response as that is shown in FIG. 11 after removing ignored frequencies.

[0069] Instead of truncating the segmented spectrum $H_s[k]$ that are not perceptible, the removal of the higher frequencies that are greater than the perceptual threshold can also be accomplished by removing the frequency spectra of the input signals. In other words, the perceptual sparse processing can be implemented after the FFT of the input signals as shown in FIG. 13(A).

[0070] Assuming that 60% of multiplications in frequency domain is removed, the number of multiplications needed for fast perceptual convolution by modifying the complexity from Algorithm 2 is calculated and illustrated in FIG. 15. From the result, the fast perceptual convolution requires about 98 real multiplications per sample to convolve with 88,200 samples of impulse response.

[0071] To evaluate the improvement in real-time systems, an experimental application has been built for evaluation. The application used two methods, the fast perceptual convolution method and Algorithm 2 respectively, to process some samples for comparison. The input block size is set to 4,096. And the test is to process single channel, 4,096×20,000 = 81,920,000 samples of input, which is about 30 minutes of samples with 44,100Hz sampling rate. The test is run on a PC with 1GHz Pentium. The result is listed in FIG. 16. As can be seen, the improved ratio is more than 30% in all cases.

[0072] Fast perceptual convolution can also be applied to the low latency implementations discussed earlier. Using the implementation shown in FIG. 10 as an example, the direct convolution part can be removed because the first 256 samples of most impulse response are belonging to the earlier delay part and the results are usually below the perceptual threshold. The implementation with fast perceptual convolution is illustrated in FIG. 17. For the impulse response "St. John Lutheran 40", the complexity can be reduced from 694 to about 324 multiplications per sample.

[0073] Although the present invention has been described with reference to the preferred embodiments, it will be understood that the invention is not limited to the details described thereof. Various substitutions and modifications have been suggested in the foregoing description, and others will occur to those of ordinary skill in the art. Therefore, all such substitutions and modifications are intended to be embraced within the scope of the invention as defined in the appended claims.